

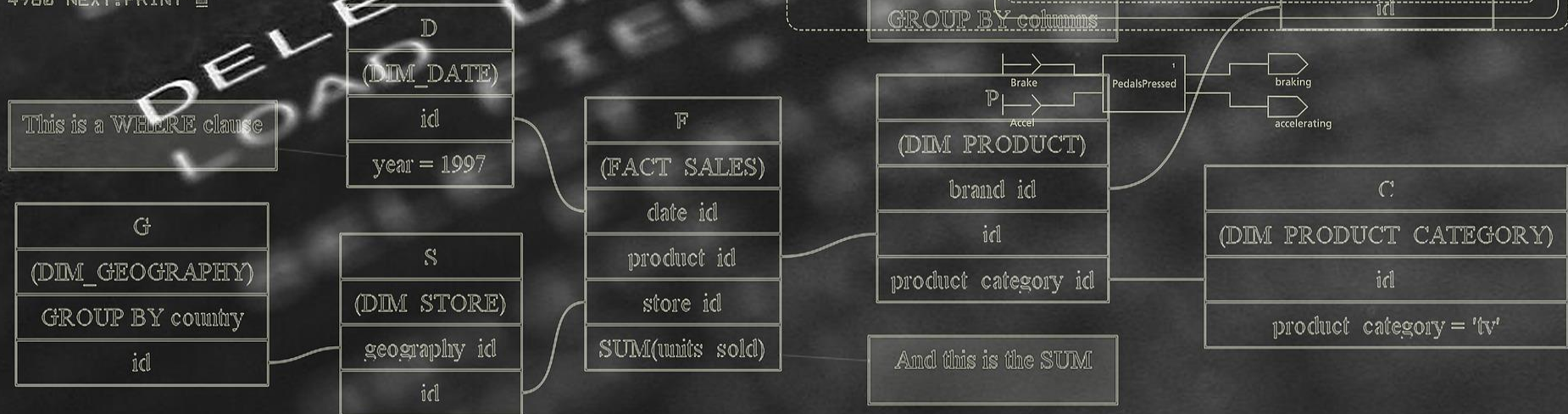
```

4780 GOTO 5000
4790 :
4800 REM
4801 REM
4802 REM
4803 REM
4810 :
4820 PRINT
4825 W=V+1
4830 FOR X
4835 FOR I
4840 PRINT
4850 NEXT:
4860 PRINT
4870 FOR I
4880 IF MD
(I+1);:GOT
4890 PRINT
4900 NEXT
4910 PRINT
4920 FOR I
4925 PRINT
4930 IF MD
";:GOTO 4
4935 PRINT
4940 NEXT:PRINT"
4950 PRINT"#####";
4960 FOR I=2 TO 24 STEP 2
4965 PRINT"|";
4970 IF MD$(I+W-1)="  " THEN PRINT" "
MB$(I)" ";:GOTO 4980
4975 PRINT MB$(I);
4980 NEXT:PRINT" "

```

SQL

RNDr. Ondřej Zýka



Obsah

- Záludnosti SQL
 - Datové typy – práce s datem
 - Příkaz Select
 - Identity
 - Null
 - Join
 - Rekurzivní with, procedurální aspekty with
 - Analytické funkce
- Problematika migračních projektů

A history of databases in No-tation

1970: NoSQL = We have no SQL

1980: NoSQL = Know SQL

2000: NoSQL = No SQL!

2005: NoSQL = Not only SQL

2013: NoSQL = No, SQL!

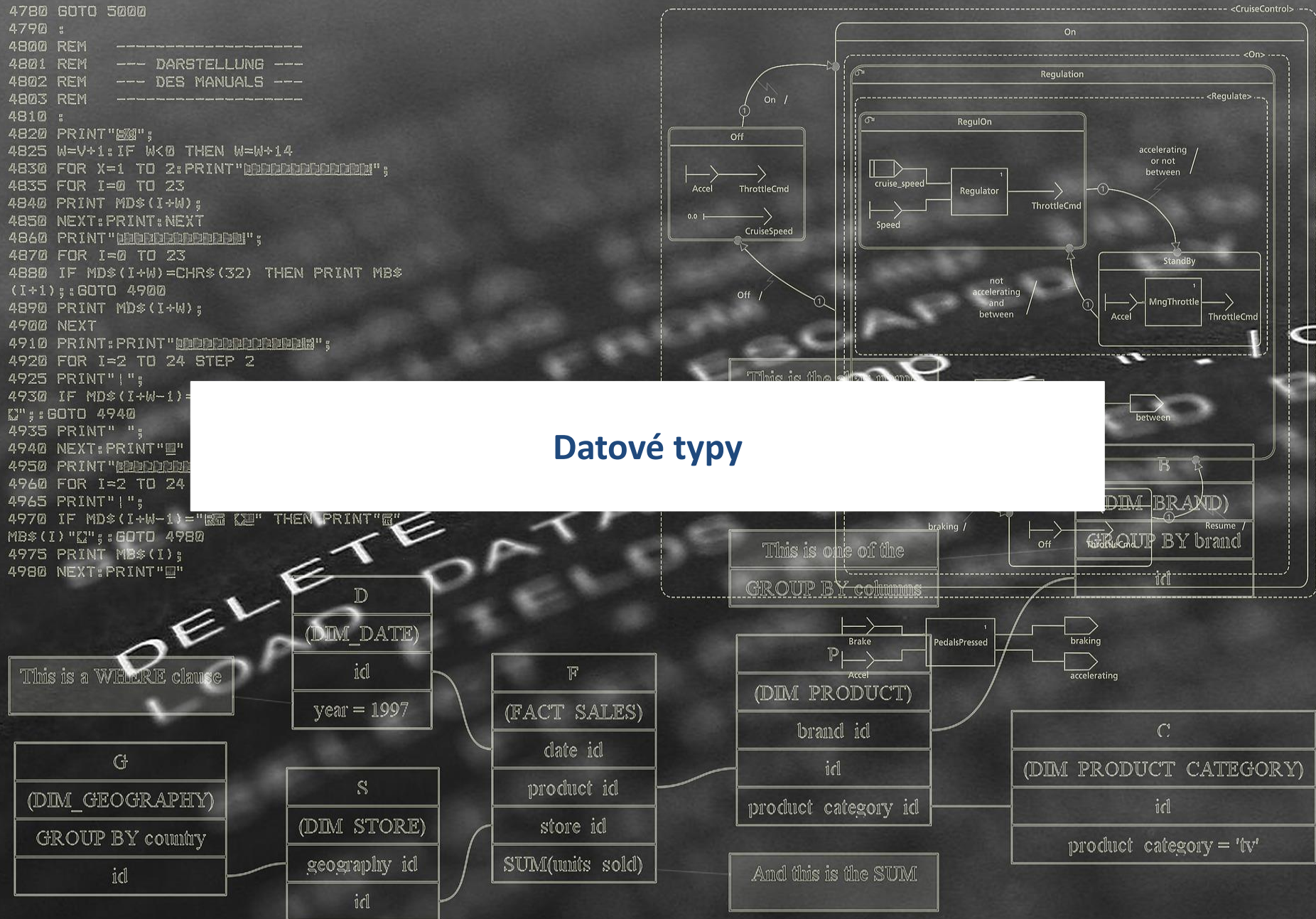
(R)DB(MS)

```

4780 GOTO 5000
4790 :
4800 REM -----
4801 REM --- DARSTELLUNG ---
4802 REM --- DES MANUALS ---
4803 REM -----
4810 :
4820 PRINT"00";
4825 W=V+1:IF W<0 THEN W=W+14
4830 FOR X=1 TO 2:PRINT"000000000000";
4835 FOR I=0 TO 23
4840 PRINT MD$(I+W);
4850 NEXT:PRINT:NEXT
4860 PRINT"000000000000";
4870 FOR I=0 TO 23
4880 IF MD$(I+W)=CHR$(32) THEN PRINT MB$
(I+1);:GOTO 4900
4890 PRINT MD$(I+W);
4900 NEXT
4910 PRINT:PRINT"000000000000";
4920 FOR I=2 TO 24 STEP 2
4925 PRINT"|";
4930 IF MD$(I+W-1)="00 00" THEN PRINT"00
";:GOTO 4940
4935 PRINT" ";
4940 NEXT:PRINT"00
4950 PRINT"000000000000";
4960 FOR I=2 TO 24
4965 PRINT"|";
4970 IF MD$(I+W-1)="00 00" THEN PRINT"00
MB$(I)"00";:GOTO 4980
4975 PRINT MB$(I);
4980 NEXT:PRINT"00

```

Datové typy



Datové typy a jejich problematické rysy

- Číselné
 - Rozsah, přesnost, interní implementace, výkonnost, velikost na disku
- Řetězce
 - Národní znaky, skutečná velikost, počáteční a koncové mezery, prázdný řetězec
- Datum a čas
 - Rozsah, aritmetika s daty, převod na a z řetězce, časové zóny, přesnost, porovnání
- Text
 - Využití v SQL, způsob uložení, operace nad textem
- Boolean
 - Efektivita, indexy
- XML
 - Množství funkcionality, efektivita použití
- Binární typy
 - Binární operace, převod na a z binární hodnoty

Microsoft datetime – datové typy

Data type	Format	Range	Accuracy	Storage size (bytes)
time	hh:mm:ss[.nnnnnnnn]	00:00:00.00000000 - 23:59:59.99999999	100 ns	3 to 5
date	YYYY-MM-DD	0001-01-01 - 9999-12-31	1 day	3
smalldatetime	YYYY-MM-DD hh:mm:ss	1900-01-01 - 2079-06-06	1 minute	4
datetime	YYYY-MM-DD hh:mm:ss[.nnn]	1753-01-01 - 9999-12-31	0.00333 s	8
datetime2	YYYY-MM-DD hh:mm:ss[.nnnnnnnn]	0001-01-01 00:00:00.00000000 - 9999-12-31 23:59:59.99999999	100 ns	6 to 8
datetimeoffset	YYYY-MM-DD hh:mm:ss[.nnnnnnnn] [+ -]hh:mm	0001-01-01 00:00:00.00000000 - 9999-12-31 23:59:59.99999999 (in UTC)	100 ns	8 to 10

Microsoft datetime - funkce

Syntax	Return data type
<code>SYSDATETIME ()</code>	<code>datetime2(7)</code>
<code>SYSDATETIMEOFFSET ()</code>	<code>datetimeoffset(7)</code>
<code>SYSUTCDATETIME ()</code>	<code>datetime2(7)</code>
<code>CURRENT_TIMESTAMP</code>	<code>datetime</code>
<code>GETDATE ()</code>	<code>datetime</code>
<code>GETUTCDATE ()</code>	<code>datetime</code>

Microsoft datetime - funkce

Syntax	Return value	Return data type
DATENAME (<i>datepart</i> , <i>date</i>)	Returns a character string that represents the specified <i>datepart</i> of the specified date.	nvarchar
DATEPART (<i>datepart</i> , <i>date</i>)	Returns an integer that represents the specified <i>datepart</i> of the specified <i>date</i> .	int
DAY (<i>date</i>)	Returns an integer that represents the day part of the specified <i>date</i> .	int
MONTH (<i>date</i>)	Returns an integer that represents the month part of a specified <i>date</i> .	int
YEAR (<i>date</i>)	Returns an integer that represents the year part of a specified <i>date</i> .	int
DATEDIFF (<i>datepart</i> , <i>startdate</i> , <i>enddate</i>)	Returns the number of date or time <i>datepart</i> boundaries that are crossed between two specified dates.	int
DATEADD (<i>datepart</i> , <i>number</i> , <i>date</i>)	Returns a new datetime value by adding an interval to the specified <i>datepart</i> of the specified <i>date</i> .	The data type of the <i>date</i> argument

Microsoft datetime - dateparts

<i>datepart</i>	Abbreviations
year	yy, yyyy
quarter	qq, q
month	mm, m
dayofyear	dy, y
day	dd, d
week	wk, ww
weekday	dw, w
hour	hh
minute	mi, n
second	ss, s
millisecond	ms
microsecond	mcs
nanosecond	ns

Microsoft datetime - convert

`CONVERT ( data_type [ ( length ) ] , expression [ , style ] )`

Without century (yy) ⁽¹⁾	With century (yyyy)	Standard	Input/Output ⁽³⁾
-	0 or 100 ^(1, 2)	Default	mon dd yyyy hh:miAM (or PM)
1	101	U.S.	mm/dd/yyyy
2	102	ANSI	yy.mm.dd
3	103	British/French	dd/mm/yyyy
4	104	German	dd.mm.yy
5	105	Italian	dd-mm-yy
6	106 ⁽¹⁾	-	dd mon yy
7	107 ⁽¹⁾	-	Mon dd, yy
8	108	-	hh:mi:ss
-	9 or 109 ^(1, 2)	Default + milliseconds	mon dd yyyy hh:mi:ss:mmmAM (or PM)

Oracle datetime – datové typy

Date type	Timezone	Fractional seconds
January 1, 4712 BC, to December 31, 9999 AD		
DATE	No	No
TIMESTAMP	No	Yes
TIMESTAMP WITH TIME ZONE	Explicit	Yes
TIMESTAMP WITH LOCAL TIME ZONE	Relative	Yes

Oracle datetime - aritmetika

```
with datum as (  
  select  
    TO_DATE('18.10.2018:23:30:14','DD.MM.YYYY:HH24:MI:SS')  
    value2,  
    TO_DATE('03.12.2004:10:34:24','DD.MM.YYYY:HH24:MI:SS')  
    value1  
  from dual  
)
```

```
SELECT  
  TO_CHAR(value2,'DD.MM.YYYY:HH24:MI:SS') "Value2",  
  TO_CHAR(value1,'DD.MM.YYYY:HH24:MI:SS') "Value1",  
  trunc(86400*(value2-value1))-60*(trunc((86400*(value2-  
    value1))/60)) "Sec",  
  trunc((86400*(value2-value1))/60)-  
  60*(trunc(((86400*(value2-  
    value1))/60)/60)) "Min",  
  trunc(((86400*(value2-value1))/60)/60)-  
    24*(trunc((((86400*(value2-value1))/60)/60)/24))  
  "Hrs",  
  trunc((((86400*(value2-value1))/60)/60)/24) "Days"  
FROM datum;
```

Oracle TIMESTAMP - aritmetika

```
with datum as (  
  select  
    TO_TIMESTAMP('18.10.2018:23:30:14', 'DD.MM.YYYY:HH24:MI  
:SS') value2,  
    TO_TIMESTAMP('03.12.2004:10:34:24', 'DD.MM.YYYY:HH24:MI  
:SS') value1  
      from dual  
)
```

```
SELECT  
  TO_CHAR(value2, 'DD.MM.YYYY:HH24:MI:SS') "Value2",  
  TO_CHAR(value1, 'DD.MM.YYYY:HH24:MI:SS') "Value1",  
  (value2-value1),  
  extract (second from (value2-value1) ) "Sec",  
  extract (minute from (value2-value1) ) "Min",  
  extract (hour from (value2-value1) ) "Hrs",  
  extract (day from (value2-value1) ) "Days"  
FROM datum;
```

Oracle datetime - funkce

- **add_months**

- `SELECT add_months(TO_DATE('28-01-2007'), 1) FROM dual;`
- `SELECT add_months(TO_DATE('30-01-2007'), 1) FROM dual;`

- **current_date**

- `SELECT sessiontimezone, current_date
FROM dual;`

- **INTERVAL '<integer>' <unit>**

- `SELECT TO_CHAR(SYSDATE + INTERVAL '10' MINUTE, 'HH:MI:SS') FROM
dual;`

- **MONTHS_BETWEEN**

- `SELECT MONTHS_BETWEEN(SYSDATE+365, SYSDATE-365) FROM dual;`

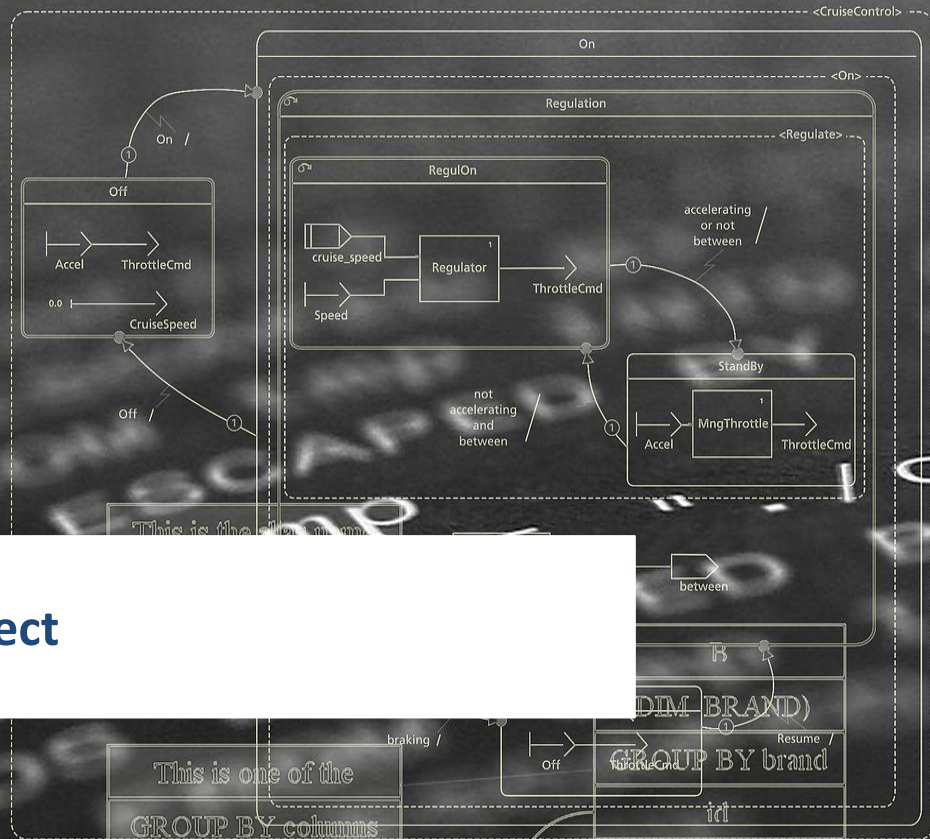
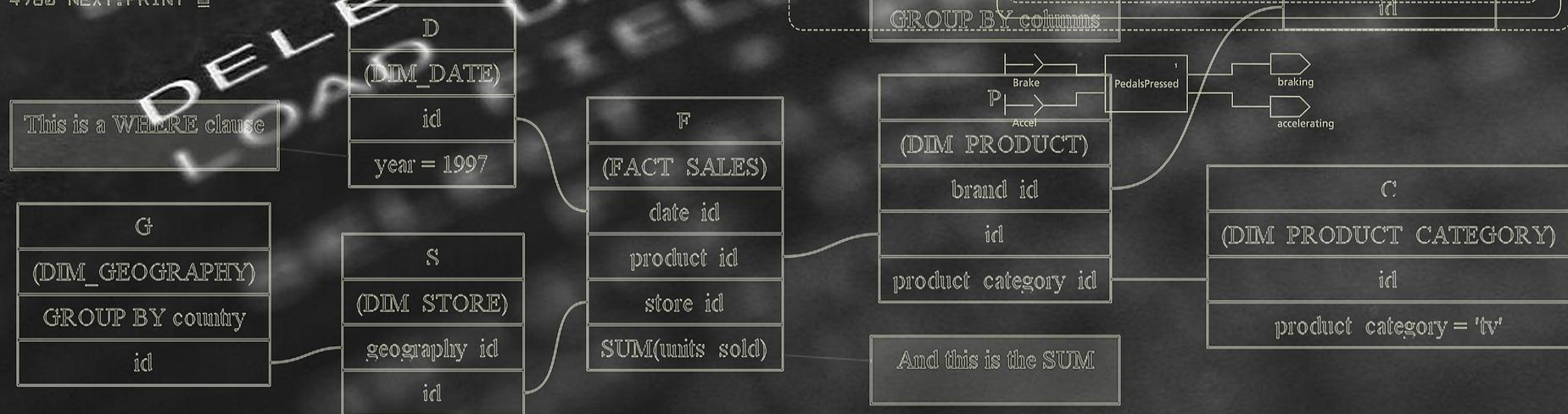
- **TO_DATE(<string1>, [format_mask], [nls_language])**

```

4780 GOTO 5000
4790 :
4800 REM -----
4801 REM --- DARSTELLUNG ---
4802 REM --- DES MANUALS ---
4803 REM -----
4810 :
4820 PRINT"00";
4825 W=V+1:IF W<0 THEN W=W+14
4830 FOR X=1 TO 2:PRINT"000000000000";
4835 FOR I=0 TO 23
4840 PRINT MD$(I+W);
4850 NEXT:PRINT:NEXT
4860 PRINT"000000000000";
4870 FOR I=0 TO 23
4880 IF MD$(I+W)=CHR$(32) THEN PRINT MB$
(I+1);:GOTO 4900
4890 PRINT MD$(I+W);
4900 NEXT
4910 PRINT:PRINT"000000000000";
4920 FOR I=2 TO 24 STEP 2
4925 PRINT"|";
4930 IF MD$(I+W-1)=
";:GOTO 4940
4935 PRINT" ";
4940 NEXT:PRINT"0"
4950 PRINT"000000000000";
4960 FOR I=2 TO 24
4965 PRINT"|";
4970 IF MD$(I+W-1)="00 00" THEN PRINT"0"
MB$(I)"0";:GOTO 4980
4975 PRINT MB$(I);
4980 NEXT:PRINT"0"

```

Select



Příkaz select – několik příkladů

select * from author

- Základní příkaz
- Mnoho modifikací a rozšíření
- Neexistují vhodná pravidla pro formátování
- Jeden příkaz x procedurální definice
- Složitost x čitelnost (upravitelnost)
- Efektivita x čitelnost (upravitelnost)
- Neexistují ustálená pravidla formátování

Příklad

```
SELECT DISTINCT DECODE(town_cd, 1, town) town, -- bud jednoznacny Town nebo null
                DECODE(district_cd, 1, district) district, -- bud jednoznacny District nebo null
                DECODE(region_cd, 1, region) region, -- bud jednoznacny Region nebo null
                zip3
INTO l_zip.town, l_zip.district, l_zip.region, l_zip.zip3
FROM (SELECT RTRIM(town) town,
            district,
            region,
            MIN(zip3) OVER() AS zip3, -- nejmensi ZIP pro danou kombinaci
            COUNT(DISTINCT town) OVER() AS town_cd, -- kardinalita Town
            COUNT(DISTINCT district) OVER() AS district_cd, -- kardinalita District
            COUNT(DISTINCT region) OVER() AS region_cd -- kardinalita Region
      FROM c_zip
     WHERE (l_s_profile.district IS NULL OR district = l_s_profile.district) -- district nevyplnen, nebo = c_zip
            AND (l_s_profile.town IS NULL OR town = l_s_profile.town) -- town nevyplnen, nebo = c_zip
            AND (l_s_profile.region IS NULL OR region = l_s_profile.region)); -- region nevyplnen, nebo = c_zip
```

select souhrnnlyi0_id as id9_62, souhrnnlyi0_datum_pocátku_platnosti as datum2_9_62, souhrnnlyi0_datum_storna as datum3_9_62, souhrnnlyi0_id_limitu as id4_9_62, souhrnnlyi0_nazev as nazev9_62, souhrnnlyi0_smlouva as smlouva9_62, nazevlimit1_klic as klic23_0, nazevlimit1_hodnota as hodnota23_0, pojisteni2_souhrnnny_limit as souhrnnny8_64, pojisteni2_id as id64, pojisteni2_id as id4_1, pojisteni2_cislo_dodatku_storna as cislo2_4_1, pojisteni2_datum_storna as datum3_4_1, pojisteni2_platnost_od as platnost4_4_1, pojisteni2_id_pojisteni as id5_4_1, pojisteni2_max_pojistne_plneni as max6_4_1, pojisteni2_nazev_id as nazev14_4_1, pojisteni2_predmet_id as predmet11_4_1, pojisteni2_sazebnik_id as sazebnik13_4_1, pojisteni2_sleva as sleva4_1, pojisteni2_smlouva_id as smlouva10_4_1, pojisteni2_souhrnnny_limit as souhrnnny8_4_1, pojisteni2_spoluucast_id as spoluucast12_4_1, pojisteni2_typ as typ4_1, nazevpoj33_klic as klic25_2, nazevpoj33_hodnota as hodnota25_2, policka4_pojisteni_id as pojisteni1_65, policko5_id as policko2_65, policko5_id as id6_3, policko5_ciselna_hodnota as ciselna2_6_3, policko5_policko as policko6_3, policko5_textova_hodnota as textova3_6_3, policko5_vyctova_hodnota as vyctova4_6_3, cpolicko6_id as id48_4, cpolicko6_typ as typ48_4, hodnoty7_policko as policko66, hodnoty7_klic as klic66, hodnoty7_klic as klic43_5, hodnoty7_hodnota as hodnota43_5, hodnoty7_policko as policko43_5, chodnotapo8_klic as klic43_6, chodnotapo8_hodnota as hodnota43_6, chodnotapo8_policko as policko43_6, predmet9_id as id7_7, predmet9_cislo_dodatku_storna as cislo2_7_7, predmet9_datum_storna as datum3_7_7, predmet9_platnost_od as platnost4_7_7, predmet9_predmet_id as predmet5_7_7, predmet9_misto_id as misto9_7_7, predmet9_nazev_predmetu as nazev11_7_7, predmet9_sazebnik as sazebnik7_7, predmet9_smlouva_id as smlouva7_7_7, predmet9_specifikace_predmetu as specifik6_7_7, predmet9_typ_pojistne_hodnoty as typ10_7_7, predmet9_typ_predmetu as typ8_7_7, predmet9_vlastnictvi_predmetu as vlastni12_7_7, mistopojis10_id as id1_8, mistopojis10_cislo_dodatku_storna as cislo2_1_8, mistopojis10_datum_storna as datum3_1_8, mistopojis10_platnost_od as platnost4_1_8, mistopojis10_misto_id as misto5_1_8, mistopojis10_cinnost_id as cinnost9_1_8, mistopojis10_popis as popis1_8, mistopojis10_zona_id as zona7_1_8, mistopojis10_smlouva_id as smlouva8_1_8, adresy11_misto_pojisteni as misto6_67, adresy11_id as id67, adresy11_id as id0_9, adresy11_psc as psc0_9, adresy11_ulice as ulice0_9, adresy11_adresa_id as adresa5_0_9, adresy11_misto_pojisteni as misto6_0_9, podnikatel12_klic as klic21_10, podnikatel12_nazev_cinnosti as nazev2_21_10, podnikatel12_odvetvi as odvetvi21_10, podnikatel13_klic as klic22_11, podnikatel13_nazev_odvetvi as nazev2_22_11, rizikovapo14_klic as klic26_12, rizikovapo14_hodnota as hodnota26_12, rizikovapo14_platnost_do as platnost3_26_12, rizikovapo14_platnost_od as platnost4_26_12, pojistnasm15_id as id5_13, pojistnasm15_cetnost_placeni as cetnost14_5_13, pojistnasm15_cislo_dodatku as cislo2_5_13, pojistnasm15_cislo_navruhu as cislo3_5_13, pojistnasm15_cislo_ps as cislo4_5_13, pojistnasm15_konec_platnosti as konec5_5_13, pojistnasm15_pocatek_platnosti as pocatek6_5_13, pojistnasm15_datum_uzavreni as datum7_5_13, pojistnasm15_korespondence as korespo18_5_13, pojistnasm15_pobocka_produkce as pobocka16_5_13, pojistnasm15_popis as popis5_13, pojistnasm15_sleva as sleva5_13, pojistnasm15stav as stav5_13, pojistnasm15_typ_pojistneho as typ17_5_13, pojistnasm15_cislo_uctu_pojistnika as cislo10_5_13, pojistnasm15_kod_banky as kod15_5_13, pojistnasm15_predcisli_uctu_pojistnika as predcisli11_5_13, pojistnasm15_specificky_symbol_uctu_pojistnika as specificky12_5_13, pojistnasm15_vysledna_pml as vysledna13_5_13, cetnostpla16_klic as klic15_14, cetnostpla16_hodnota as hodnota15_14, koresponde17_klic as klic20_15, koresponde17_hodnota as hodnota20_15, osoby18_smlouva_id as smlouva11_68, osoby18_id as id68, osoby18_id as id3_16, osoby18_adresa_id as adresa10_3_16, osoby18_koresp_adresa_id as koresp14_3_16, osoby18_cislo_pasu as cislo2_3_16, osoby18_evidencni_vypis as evidencni3_3_16, osoby18_funkce as funkce3_16, osoby18_ico as ico3_16, osoby18_jmeno as jmeno3_16, osoby18_nazev_firmy as nazev7_3_16, osoby18_prijmeni as prijmeni3_16, osoby18_rodne_cislo as rodne9_3_16, osoby18_role as role3_16, osoby18_smlouva_id as smlouva11_3_16, osoby18_statni_prislusnost as statni15_3_16, osoby18_titul_id as titul13_3_16, osoby18_typ as typ3_16, adresa19_id as id0_17, adresa19_psc as psc0_17, adresa19_ulice as ulice0_17, adresa19_adresa_id as adresa5_0_17, adresa19_misto_pojisteni as misto6_0_17, adresa19_typ as typ0_17, adresa20_id as id0_18, adresa20_psc as psc0_18, adresa20_ulice as ulice0_18, adresa20_adresa_id as adresa5_0_18, adresa20_misto_pojisteni as misto6_0_18, adresa20_typ as typ0_18, roleosoby21_klic as klic27_19, roleosoby21_hodnota as hodnota27_19, statnipris22_klic as klic29_20, statnipris22_hodnota as hodnota29_20, titul23_klic as klic31_21, titul23_hodnota as hodnota31_21, typosoby24_klic as klic33_22, typosoby24_hodnota as hodnota33_22, cislopoboc25_klic as klic16_23, cislopoboc25_hodnota as hodnota16_23, pojisteni26_smlouva_id as smlouva10_69, pojisteni26_id as id69, pojisteni26_id as id4_24, pojisteni26_cislo_dodatku_storna as cislo2_4_24, pojisteni26_datum_storna as datum3_4_24, pojisteni26_platnost_od as platnost4_4_24, pojisteni26_id_pojisteni as id5_4_24, pojisteni26_max_pojistne_plneni as max6_4_24, pojisteni26_nazev_id as nazev14_4_24, pojisteni26_predmet_id as predmet11_4_24, pojisteni26_sazebnik_id as sazebnik13_4_24, pojisteni26_sleva as sleva4_24, pojisteni26_smlouva_id as smlouva10_4_24, pojisteni26_souhrnnny_limit as souhrnnny8_4_24, pojisteni26_spoluucast_id as spoluucast12_4_24, pojisteni26_typ as typ4_24, csazebnik27_id as id50_25, csazebnik27_nazev as nazev50_25, csazebnik27_platnost_do as platnost3_50_25, csazebnik27_platnost_od as platnost4_50_25, predmetyap28_sazebnik as sazebnik70, predmetyap28_platnost_do as platnost2_70, predmetyap28_platnost_od as platnost3_70, predmetyap28_pojisteni as pojisteni70, predmetyap28_predmet as predmet70, cpojisteni29_id as id47_26, cpojisteni29_nazev as nazev47_26, konverze30_pojisteni_id as pojisteni5_71, konverze30_id as id71, konverze30_id as id44_27, konverze30_id_oj as id2_44_27, konverze30_platnost_od as platnost3_44_27, konverze30_platnost_od as platnost4_44_27, konverze30_pojisteni_id as pojisteni5_44_27, okamzikyaa31_pojisteni_id as pojisteni1_72, okamzikyaa31_akce_id as akce2_72, okamzikyaa31_okamzik_id as okamzik3_72, okamzikyaa31_poradi as poradi72, okamzikyaa31_platnost_do as platnost5_72, okamzikyaa31_platnost_od as platnost6_72, cakce32_id as id41_28, cakce32_java_program as java2_41_28, cokamzik33_id as id45_29, cokamzik33_nazev as nazev45_29, policka34_pojisteni_id as pojisteni1_73, policka34_platnost_do as platnost2_73, policka34_platnost_od as platnost3_73, policka34_policko_id as policko4_73, cpolicko35_id as id48_30, cpolicko35_typ as typ48_30, souhrnnli36_pojisteni_id as pojisteni1_74, csouhrnnly37_id as souhrnnne2_74, csouhrnnly37_id as id51_31, csouhrnnly37_nazev as nazev51_31, cpredmet38_id as id49_32, cpredmet38_misto_pojivninne as misto2_49_32, cpredmet38_nazev as nazev49_32, cpredmet38_typ as typ49_32, typpredmet39_klic as klic37_33, typpredmet39_hodnota as hodnota37_33, spoluucast40_klic as klic28_34, spoluucast40_hodnota as hodnota28_34, ctyppojst41_klic as klic34_35, ctyppojst41_hodnota as hodnota34_35, otazky42_pojisteni_id as pojisteni1_75, otazky42_otazka_id as otazka2_75, otazky42_platnost_do as platnost3_75, otazky42_platnost_od as platnost4_75, cotazka43_id as id46_36, cotazka43_poradi as poradi46_36, cotazka43_typ as typ46_36, zavisina44_pojisteni_id as pojisteni1_76, zavisina44_master_id as master2_76, zavisina44_pozadovany_vysledek as pozadovany3_76, cotazka45_id as id46_37, cotazka45_poradi as poradi46_37, cotazka45_typ as typ46_37, predmety46_smlouva_id as smlouva7_77, predmety46_id as id77, predmety46_id as id7_38, predmety46_cislo_dodatku_storna as cislo2_7_38, predmety46_datum_storna as datum3_7_38, predmety46_platnost_od as platnost4_7_38, predmety46_predmet_id as predmet5_7_38, predmety46_misto_id as misto9_7_38, predmety46_nazev_predmetu as nazev11_7_38, predmety46_sazebnik as sazebnik7_38, predmety46_smlouva_id as smlouva7_7_38, predmety46_specifikace_predmetu as specifik6_7_38, predmety46_typ_pojistne_hodnoty as typ10_7_38, predmety46_typ_predmetu as typ8_7_38, predmety46_vlastnictvi_predmetu as vlastni12_7_38, nazevpredm47_klic as klic24_39, nazevpredm47_hodnota as hodnota24_39, nemovitost48_predmet as predmet78, nemovitost48_id as id78, nemovitost48_id as id2_40, nemovitost48_cena as cena2_40, nemovitost48_index as index2_40, nemovitost48_predmet as predmet2_40, nemovitost48_psc as psc2_40, nemovitost48_ulice as ulice2_40, nemovitost48_specifikace as specifik7_2_40, policka49_predmet_id as predmet1_79, policko50_id as policko2_79, policko50_id as id6_41, policko50_ciselna_hodnota as ciselna2_6_41, policko50_policko as policko6_41, policko50_textova_hodnota as textova3_6_41, policko50_vyctova_hodnota as vyctova4_6_41, csazebnik51_id as id50_42, csazebnik51_nazev as nazev50_42, csazebnik51_platnost_do as platnost3_50_42, csazebnik51_platnost_od as platnost4_50_42, typpojst52_klic as klic36_43, typpojst52_hodnota as hodnota36_43, typpredmet53_klic as klic37_44, typpredmet53_hodnota as hodnota37_44, vlastnictv54_klic as klic39_45, vlastnictv54_hodnota as hodnota39_45, vozidla55_predmet as predmet80, vozidla55_vozidlo_id as vozidlo1_80, vozidla55_vozidlo_id as vozidlo1_12_46, vozidla55_cena as cena12_46, vozidla55_index as index12_46, vozidla55_predmet as predmet12_46, vozidla55_cena_nova as cena4_12_46, vozidla55_cislo_karoserie as cislo5_12_46, vozidla55_druh as druh12_46, vozidla55_registracni_znacka as registra6_12_46, vozidla55_rok_vyroby as rok7_12_46, vozidla55_typ_provedeni as typ9_12_46, vozidla55_znacka as znacka12_46, druhozidil56_klic as klic18_47, druhozidil56_hodnota as hodnota18_47, typprovede57_klic as klic38_48, typprovede57_hodnota as hodnota38_48, znackavozis58_klic as klic40_49, znackavozis58_hodnota as hodnota40_49, zarizeni59_predmet as predmet81, zarizeni59_zarizeni_id as zarizeni1_81, zarizeni59_zarizeni_id as zarizeni13_50, zarizeni59_cena as cena13_50, zarizeni59_index as index13_50, zarizeni59_predmet as predmet13_50, zarizeni59_rok_vyroby as rok4_13_50, zarizeni59_specifikace as specifik5_13_50, zarizeni59_typ as typ13_50, zarizeni59_vyrobní_cislo as vyrobní7_13_50, prilohy60_smlouva_id as smlouva4_82, prilohy60_id as id82, prilohy60_id as id8_51, prilohy60_druh as druh8_51, prilohy60_id_prilohy as id2_8_51, prilohy60_pocet_stran as pocet3_8_51, prilohy60_smlouva_id as smlouva4_8_51, druhriloh61_klic as klic17_52, druhriloh61_hodnota as hodnota17_52, souhrnnli62_smlouva as smlouva83, souhrnnli62_id as id83, souhrnnli62_id as id9_53, souhrnnli62_datum_pocátku_platnosti as datum2_9_53, souhrnnli62_datum_storna as datum2_9_53, souhrnnli62_id_limitu as id4_9_53, souhrnnli62_nazev as nazev9_53, souhrnnli62_smlouva as smlouva9_53, specialniu63_smlouva_id as smlouva3_84, specialniu63_id as id84, specialniu63_id as id10_54, specialniu63_smlouva_id as smlouva3_10_54, specialniu63_text as text10_54, spravci64_smlouva_id as smlouva1_85, spravce65_id as spravce2_85, spravce65_id as id11_55, spravce65_cislo_pobocky as cislo8_11_55, spravce65_cislo_spravce as cislo2_11_55, spravce65_email as email11_55, spravce65_jmeno as jmeno11_55, spravce65_osobni_cislo as osobni5_11_55, spravce65_prijmeni as prijmeni11_55, spravce65_telefon as telefon11_55, cislopoboc66_klic as klic16_56, cislopoboc66_hodnota as hodnota16_56, stavsmlouv67_klic as klic30_57, stavsmlouv67_hodnota as hodnota30_57, typpojst68_klic as klic35_58, t

Příkaz select

select

city,

count(*)

from author

where city like '%o%'

group by city

having count(*) > 1

order by city

- Základní části příkazu
- Funkce
- Třídění
- Jména tabulek
- Case sensitive/insensitive

Příkaz select

select

l_name,
NumOfAuthors

from author,

(select
city,
count(*) as NumOfAuthors

from author

group by city) CityCount

where

author.city =
CityCount.city

and CityCount.NumOfAuthors
> 1

order by author.f_name

- Select *
- Odvozené tabulky (Derived tables)
- „Klasický“ join

Příkaz select

```
with CityCount as

  (select
    city,
    count(*) as NumOfAuthors
  from author
  group by city)

select

  l_name,
  NumOfAuthors

from author join CityCount on
  (author.city = CityCount.city)

where CityCount.NumOfAuthors > 1

order by author.f_name desc
```

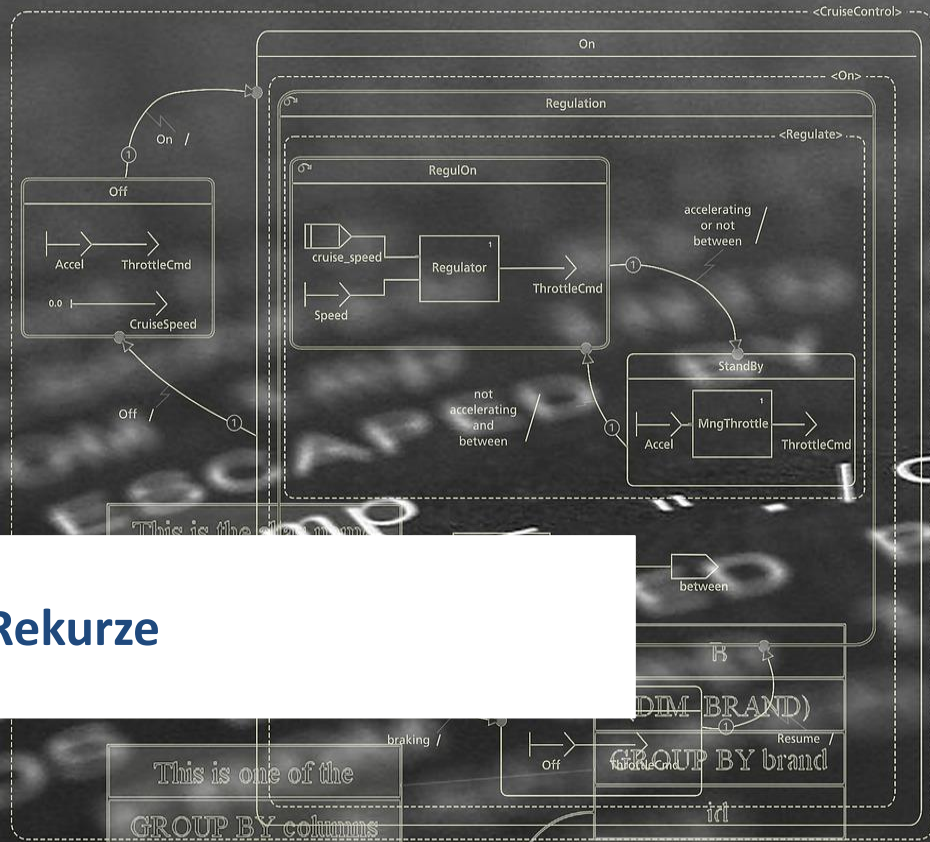
- With
- Kvalifikovaná jména (dbo)
- Přepsání jména sloupců
- Unikátnost jmen sloupců
- Třídění
- Ansi join

```

4780 GOTO 5000
4790 :
4800 REM -----
4801 REM --- DARSTELLUNG ---
4802 REM --- DES MANUALS ---
4803 REM -----
4810 :
4820 PRINT"00";
4825 W=V+1:IF W<0 THEN W=W+14
4830 FOR X=1 TO 2:PRINT"000000000000";
4835 FOR I=0 TO 23
4840 PRINT MD$(I+W);
4850 NEXT:PRINT:NEXT
4860 PRINT"000000000000";
4870 FOR I=0 TO 23
4880 IF MD$(I+W)=CHR$(32) THEN PRINT MB$
(I+1);:GOTO 4900
4890 PRINT MD$(I+W);
4900 NEXT
4910 PRINT:PRINT"000000000000";
4920 FOR I=2 TO 24 STEP 2
4925 PRINT"|";
4930 IF MD$(I+W-1)="00 00" THEN PRINT"00
";:GOTO 4940
4935 PRINT" ";
4940 NEXT:PRINT"00"
4950 PRINT"000000000000";
4960 FOR I=2 TO 24
4965 PRINT"|";
4970 IF MD$(I+W-1)="00 00" THEN PRINT"00
MB$(I)"00";:GOTO 4980
4975 PRINT MB$(I);
4980 NEXT:PRINT"00"

```

With a Rekurze



With – strukturované programování

```
with
title_ext as (
select title.title_id, title||' ('|| LISTAGG(l_name, ',') WITHIN GROUP
(ORDER BY l_name)||')' as title_desc
from author
        join title_author on (title_author.au_id = author.au_id)
        join title on (title_author.title_id = title.title_id)
group by title.title_id,title
)
--select * from title_ext;

,sales_ext as (
select sum(qty) as qty, name , title_id from
        sales_detail
        join store on (store.store_id = sales_detail.store_id)
group by name, title_id
)
--select * from sales_ext;

select title_desc, name, qty
from title_ext
        join sales_ext on (title_ext.title_id = sales_ext.title_id);
```

Rekurze

- Oracle
- „Klasická“ konstrukce pomocí connect
- Ještě před ANSI definicí

```
SELECT ROWNUM AS ID  
FROM dual  
CONNECT BY LEVEL <= 10
```

- Rekurzivní with
- ANSI norma, Microsoft, Oracle

```
with x (id) as  
(select 1  
 union all  
  select id+1 from x  
  where id<10)  
select * from x
```

Rekurze

Vytvořte tabulku s hodnotami
1 až 100

ID

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
...

Řešení

with

numbers (val) as

(**select** 1 as val **from** dual

union all

select val+1 **from** numbers **where** val < 100)

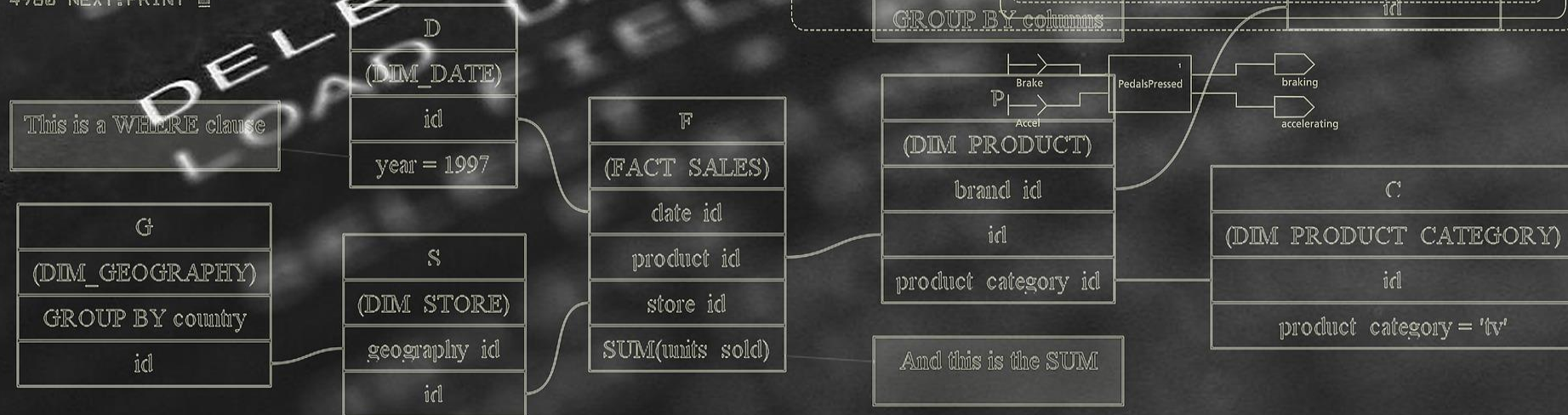
select val as ID **from** numbers;

```

4780 GOTO 5000
4790 :
4800 REM -----
4801 REM --- DARSTELLUNG ---
4802 REM --- DES MANUALS ---
4803 REM -----
4810 :
4820 PRINT" ";
4825 W=V+1:IF W<0 THEN W=W+14
4830 FOR X=1 TO 2:PRINT" ";
4835 FOR I=0 TO 23
4840 PRINT MD$(I+W);
4850 NEXT:PRINT:NEXT
4860 PRINT" ";
4870 FOR I=0 TO 23
4880 IF MD$(I+W)=CHR$(32) THEN PRINT MB$
(I+1);:GOTO 4900
4890 PRINT MD$(I+W);
4900 NEXT
4910 PRINT:PRINT" ";
4920 FOR I=2 TO 24 STEP 2
4925 PRINT"|";
4930 IF MD$(I+W-1)=
";:GOTO 4940
4935 PRINT" ";
4940 NEXT:PRINT" "
4950 PRINT" ";
4960 FOR I=2 TO 24
4965 PRINT"|";
4970 IF MD$(I+W-1)=" " THEN PRINT" "
MB$(I)" ";:GOTO 4980
4975 PRINT MB$(I);
4980 NEXT:PRINT" "

```

Identity - sekvence



Identity/Sekvence

- Generování číselného umělého klíče
- Identity/Sekvence
- Potřeba práce mimo transakce
- Požadavek vysokého výkonu
- Alokace bloků, výpadky při pádu serveru nebo v clusteru.
- Sekvence umožňují globální i lokální číslování objektů
- Identity – speciální typ sloupce, složitější administrace
- Sequence – další objekt v databázi, složitější administrace
- Nedá se zaručit generování nepřerušená sekvence

Oracle - Sequence

```
CREATE SEQUENCE SQ_TEST  
    INCREMENT BY 1  
    START WITH 1;
```

```
CREATE TABLE T_TEST (ID number, tx varchar (100));
```

```
select SQ_TEST.nextval from dual;
```

```
select SQ_TEST.currval from dual;
```

```
insert into T_TEST values (SQ_TEST.nextval, 'text');
```

Microsoft - Identity

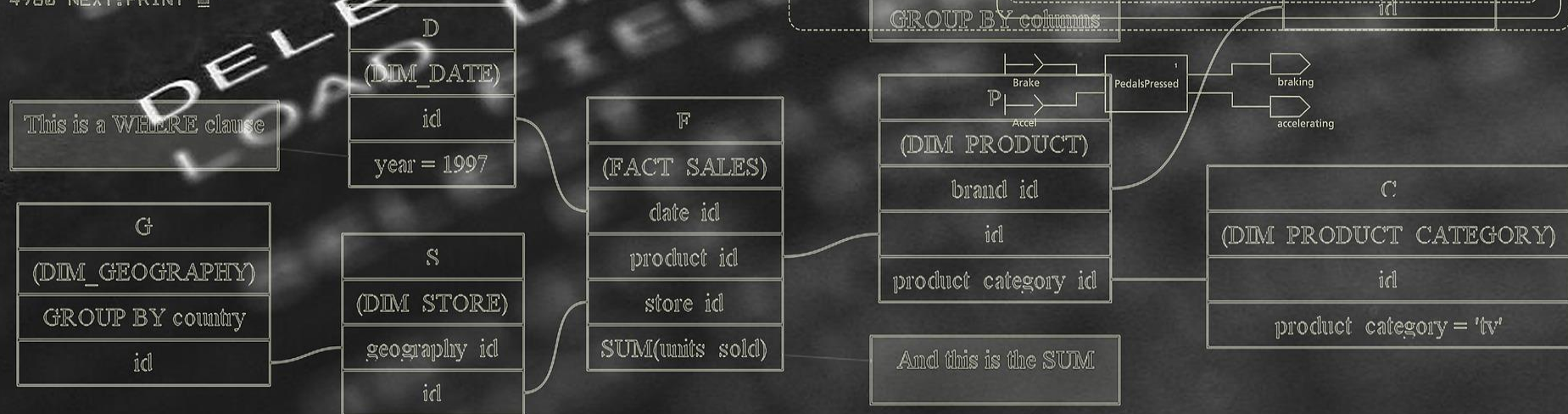
```
create table T_TEST  
(id numeric(10,0) identity,  
  tx varchar(100)  
)  
  
insert T_TEST values ('text')
```

```

4780 GOTO 5000
4790 :
4800 REM -----
4801 REM --- DARSTELLUNG ---
4802 REM --- DES MANUALS ---
4803 REM -----
4810 :
4820 PRINT" ";
4825 W=V+1:IF W<0 THEN W=W+14
4830 FOR X=1 TO 2:PRINT" ";
4835 FOR I=0 TO 23
4840 PRINT MD$(I+W);
4850 NEXT:PRINT:NEXT
4860 PRINT" ";
4870 FOR I=0 TO 23
4880 IF MD$(I+W)=CHR$(32) THEN PRINT MB$
(I+1);:GOTO 4900
4890 PRINT MD$(I+W);
4900 NEXT
4910 PRINT:PRINT" ";
4920 FOR I=2 TO 24 STEP 2
4925 PRINT"|";
4930 IF MD$(I+W-1)=
";:GOTO 4940
4935 PRINT" ";
4940 NEXT:PRINT" "
4950 PRINT" ";
4960 FOR I=2 TO 24
4965 PRINT"|";
4970 IF MD$(I+W-1)=" " THEN PRINT" "
MB$(I)" ";:GOTO 4980
4975 PRINT MB$(I);
4980 NEXT:PRINT" "

```

NULL



Null

```
select title_id from title where price = null
```

title_id	title	type	pub_id	price	advance	total_sales
MC3026	The Psychology of Computer Cooking	psychology	0877	(null)	(null)	2606
PC9999	Net Etiquette	popular_comp	1389	(null)	(null)	3528

```
title_id
-----
MC3026
PC9999

(2 row(s) affected)
```

```
title_id
-----

(0 row(s) affected)
```

Null

```
set ANSI_NULLS on  
set ANSI_NULLS off
```

- Použití null s = je programátorská chyba
- Záleží na kontextu
- Chování může být mimo vliv programátora

```
select title_id from title where price is null  
select title_id from title where price is not null
```

Příklad

```
CREATE TABLE colors (  
    color    VARCHAR(10) NOT NULL PRIMARY KEY  
);
```

```
INSERT INTO colors VALUES ( 'Blue' );  
INSERT INTO colors VALUES ( 'Red' );  
INSERT INTO colors VALUES ( 'Green' );  
INSERT INTO colors VALUES ( 'Black' );
```

```
CREATE TABLE products (  
    product_id          NUMBER NOT NULL PRIMARY KEY,  
    product_description  VARCHAR(30) NOT NULL,  
    color                VARCHAR(10)  
);
```

```
INSERT INTO products VALUES(1, 'Ball', 'Red');  
INSERT INTO products VALUES(2, 'Bike', 'Blue');  
INSERT INTO products VALUES(3, 'Tent', NULL);
```

Najdi nepoužité barvy

```
SELECT c.color  
FROM colors c  
WHERE c.color NOT IN (SELECT p.color FROM products p) ;
```

no rows selected

Řešení

- `/* EXISTS */`
- `SELECT c.color`
- `FROM colors c`
- `WHERE NOT EXISTS (SELECT * FROM products p WHERE
c.color = p.color);`

`/* IS NOT NULL in the subquery */`

```
SELECT c.color
FROM colors c
WHERE c.color NOT IN (SELECT p.color FROM products p
                      WHERE p.color IS NOT NULL
                      );
```

Řešení

- `/* MINUS */`
- `SELECT color FROM colors`
- `MINUS`
- `SELECT color FROM products;`

`/* LEFT OUTER JOIN */`

```
SELECT c.color
FROM colors c
      LEFT OUTER JOIN products p ON c.color = p.color
WHERE
      p.color IS NULL;
```

Dotaz na null hodnotu v kódu

```
CREATE OR REPLACE FUNCTION get_count_1 (  
    p_color VARCHAR  
) RETURN NUMBER AS  
    l_num    NUMBER;  
BEGIN  
    SELECT COUNT(*)  
    INTO l_num  
    FROM products  
    WHERE color = p_color;  
    RETURN l_num;  
END;  
/
```

```
SELECT get_count_1('Red') FROM dual;
```

```
1
```

```
SELECT get_count_1(null) FROM dual;
```

```
0
```

Řešení

```
CREATE OR REPLACE FUNCTION get_count_2 (  
    p_color VARCHAR  
) RETURN NUMBER AS  
    l_num    NUMBER;  
BEGIN  
    IF p_color IS NULL THEN  
        SELECT COUNT(*) INTO l_num FROM products WHERE  
            color IS NULL;  
    ELSE  
        SELECT COUNT(*) INTO l_num FROM products WHERE  
            color = p_color;  
    END IF;  
    RETURN l_num;  
END;  
/  
  
SELECT get_count_2(NULL) FROM dual;  
  
SELECT get_count_2('Red') FROM dual;
```

Jiné řešení

```
CREATE OR REPLACE FUNCTION get_count_3 (  
    p_color VARCHAR  
) RETURN NUMBER AS  
    l_num    NUMBER;  
BEGIN  
    SELECT COUNT(*) INTO l_num FROM products  
        WHERE NVL(color, 'N/A') = NVL(p_color, 'N/A');  
    RETURN l_num;  
END;  
/
```

```
SELECT get_count_3(NULL) FROM dual;
```

```
SELECT get_count_3('Red') FROM dual;
```

Porovnání řádků

```
create or replace FUNCTION compare_products_1(  
    p_product_id_1 NUMBER,  
    p_product_id_2 NUMBER  
) RETURN NUMBER AS  
    l_row1 products%ROWTYPE;  
    l_row2 products%ROWTYPE;  
BEGIN  
    select * into l_row1 from products where product_id =  
p_product_id_1;  
    select * into l_row2 from products where product_id =  
p_product_id_2;  
    if l_row1.product_description =  
l_row2.product_description and l_row1.color = l_row2.color  
then  
        return 0;  
    else  
        return 1;  
    end if;  
END;
```

Porovnávání null

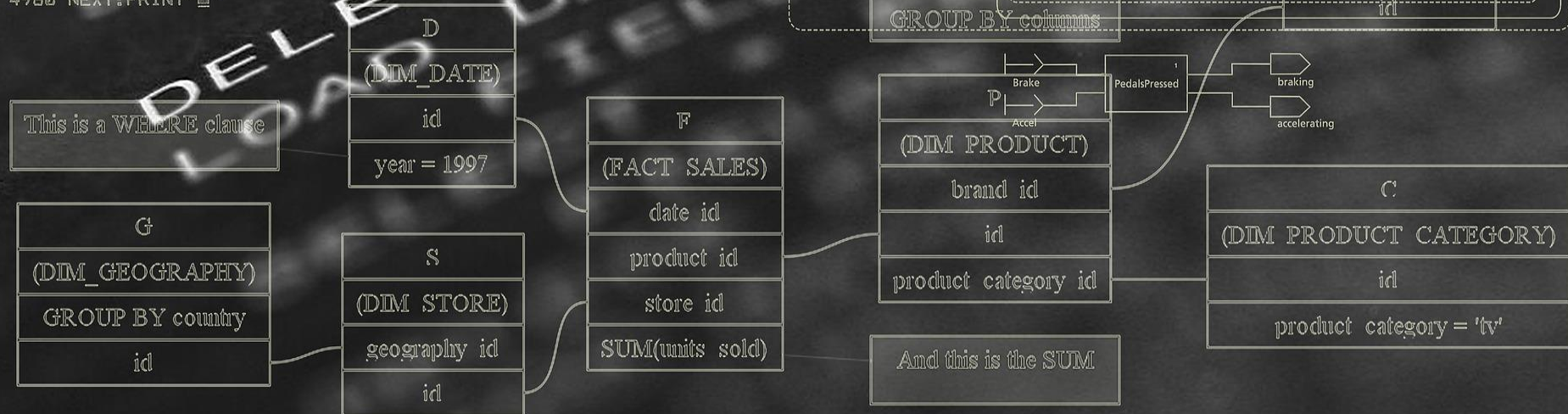
- Null **není** jedna speciální hodnota (jako například v java nebo C)
- Nutno rozdělit případy kdy se může vyskytnout při porovnání null hodnota a kdy ne
- Dá se použít konstrukce
`NVL(color, 'N/A') = NVL(p_color, 'N/A');`
- Je nutné zvolit vhodnou nevýznamovou hodnotu ('N/A') .
 - Jde pro dlouhé řetězce
 - Jde pro datum
 - Nejde pro krátké řetězce (všechny hodnoty mohou být významové)
 - Nejde pro čísla
- Vždy jde použít
`( (color = p_color) or (color is null and p_color is null) )`

```

4780 GOTO 5000
4790 :
4800 REM -----
4801 REM --- DARSTELLUNG ---
4802 REM --- DES MANUALS ---
4803 REM -----
4810 :
4820 PRINT"00";
4825 W=V+1:IF W<0 THEN W=W+14
4830 FOR X=1 TO 2:PRINT"000000000000";
4835 FOR I=0 TO 23
4840 PRINT MD$(I+W);
4850 NEXT:PRINT:NEXT
4860 PRINT"000000000000";
4870 FOR I=0 TO 23
4880 IF MD$(I+W)=CHR$(32) THEN PRINT MB$
(I+1);:GOTO 4900
4890 PRINT MD$(I+W);
4900 NEXT
4910 PRINT:PRINT"000000000000";
4920 FOR I=2 TO 24 STEP 2
4925 PRINT"|";
4930 IF MD$(I+W-1)=
";:GOTO 4940
4935 PRINT" ";
4940 NEXT:PRINT"0"
4950 PRINT"000000000000";
4960 FOR I=2 TO 24
4965 PRINT"|";
4970 IF MD$(I+W-1)="00 00" THEN PRINT"0"
MB$(I)"0";:GOTO 4980
4975 PRINT MB$(I);
4980 NEXT:PRINT"0"

```

Join



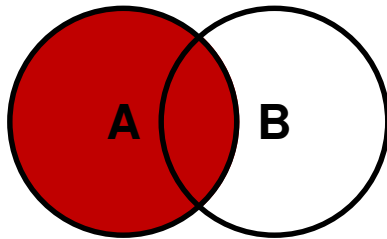
Join null hodnot

```
select a.title_id, b.title_id
from title a, title b
where a.price = b.price
```

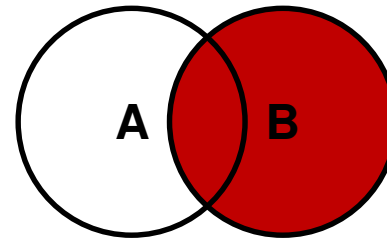
- Ani jeden řádek s price is null

```
... join on (a.price = b.price)
... join on (a.price = b. price or (a. price is null
    and b. price is null) )
... join on (nvl(a.price,'N/A') = nvl(b.price,'N/A'))
```

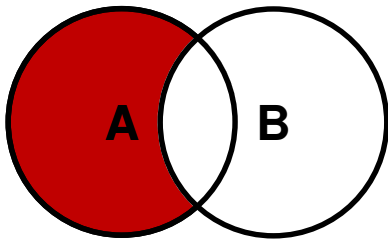
Join



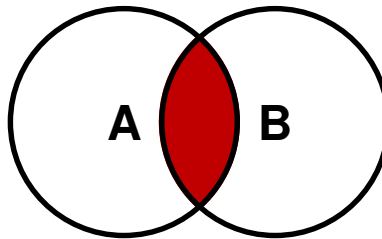
```
select * from  
A left join B  
on A.PK = B.PK
```



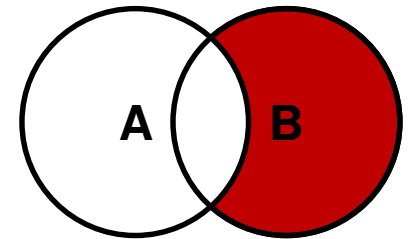
```
select * from  
A right join B  
on A.PK = B.PK
```



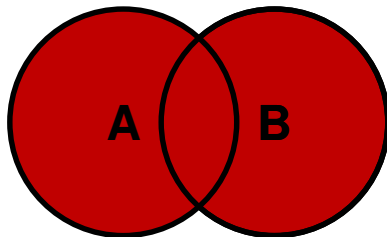
```
select * from  
A left join B  
on A.PK = B.PK  
where B.PK is null
```



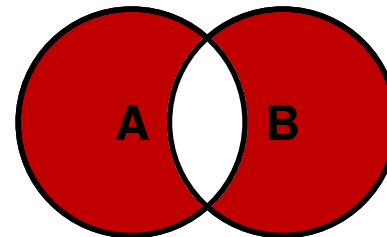
```
select * from  
A inner join B  
on A.PK = B.PK
```



```
select * from  
A right join B  
on A.PK = B.PK  
where A.PK is null
```



```
select * from  
A full outer join B  
on A.PK = B.PK
```



```
select * from  
A full outer join B  
on A.PK = B.PK  
where A.PK is null  
or B.PK is null
```

Join

```
select store.store_id, sale.ord_num  
from store, sale  
where store.store_id = sale.store_id  
and sale.ord_date = '01.01.2010'
```

- Klasická notace
- Definice přes kartézský součin a podmínky ve where
- Nedoporučovaný zápis – mladí programátoři už zápisu nerozumí

Join

```
select store.store_id, sale.ord_num from  
store join sale  
on store.store_id = sale.store_id  
where sale.ord_date = '01.01.2010'
```

```
select store.store_id, sale.ord_num from  
store join sale  
on (store.store_id = sale.store_id  
and sale.ord_date = '01.01.2010')
```

- Ansi notace
- Přesně definovaný výsledek operace join

Join

```
select l_name, title from  
title join title_author join author  
    on (title_author.au_id = author.au_id)  
    on (title.title_id = title_author.title_id)
```

```
select l_name, title from  
title join title_author  
on (title.title_id = title_author.title_id)  
    join author  
on (title_author.au_id = author.au_id)
```

- **Ansi notace**
 - Definuje pořadí spojení
 - Definuje kdy vyhodnocovat podmínky
- Podstatné u databází, které neoptimalizují pořadí v join operaci

Outer join

```
select store.store_id, count(sale.ord_num)
from store, sale
where store.store_id *= sale.store_id
and sale.ord_date = '2010-01-01'
group by store.store_id
order by 2
```

- Microsoft, Sybase – stará (klasická) notace
- Definice přes kartézský součin

Outer join

```
select store.store_id, count(sale.ord_num) from
store, sale
where store.store_id = sale.store_id (+)
and sale.ord_date(+) = TO_DATE('01.01.2010','DD.MM.YYYY')
group by store.store_id
Order by 2,1;
```

```
select store.store_id, count(dd.ord_num)
from store,
    (select store_id, ord_date, ord_num
     from sale where ord_date = TO_DATE('01.01.2010','DD.MM.YYYY')) dd
where store.store_id = dd.store_id (+)
group by store.store_id
order by 2,1;
```

Oracle používá jinou notaci a semantiku, (+) u datumu je nutné

Outer join

```
select store.store_id, count(sale.ord_num) from
store left outer join sale
on (store.store_id = sale.store_id
    and sale.ord_date = TO_DATE('01.01.2010','DD.MM.YYYY'))
group by store.store_id
order by 2,1;
```

ANSI notace funguje

Datum musí být v podmínce

Chybně:

```
select store.store_id, count(sale.ord_num) from
store right outer join sale on (store.store_id = sale.store_id)
where sale.ord_date = TO_DATE('01.01.2010','DD.MM.YYYY')
group by store.store_id, sale.ord_date
order by 2,1;
```

Outer join - shrnutí

- Klasická notace
 - Liší se u jednotlivých databází
 - Mnoho kódu používá klasickou notaci
 - Hodně uživatelů zná klasickou notaci
 - !!! Není jednoznačná
 - Některé nástroje ji už nepodporují
 - Často čitelnější než ANSI notace

Join - shrnutí

- ANSI notace
 - Jednoznačná
 - Mnoho typů joinů
 - [Inner] Join
 - [Left | right] outer join
 - Full join
 - Cross join
 - !!!! Natural Join
 - Rozdílná podpora u různých dodavatelů

Join - shrnutí

- Podmínka joinu
 - =
 - <
 - Jiná
- Selfjoin
- Join velkého počtu tabulek

Select

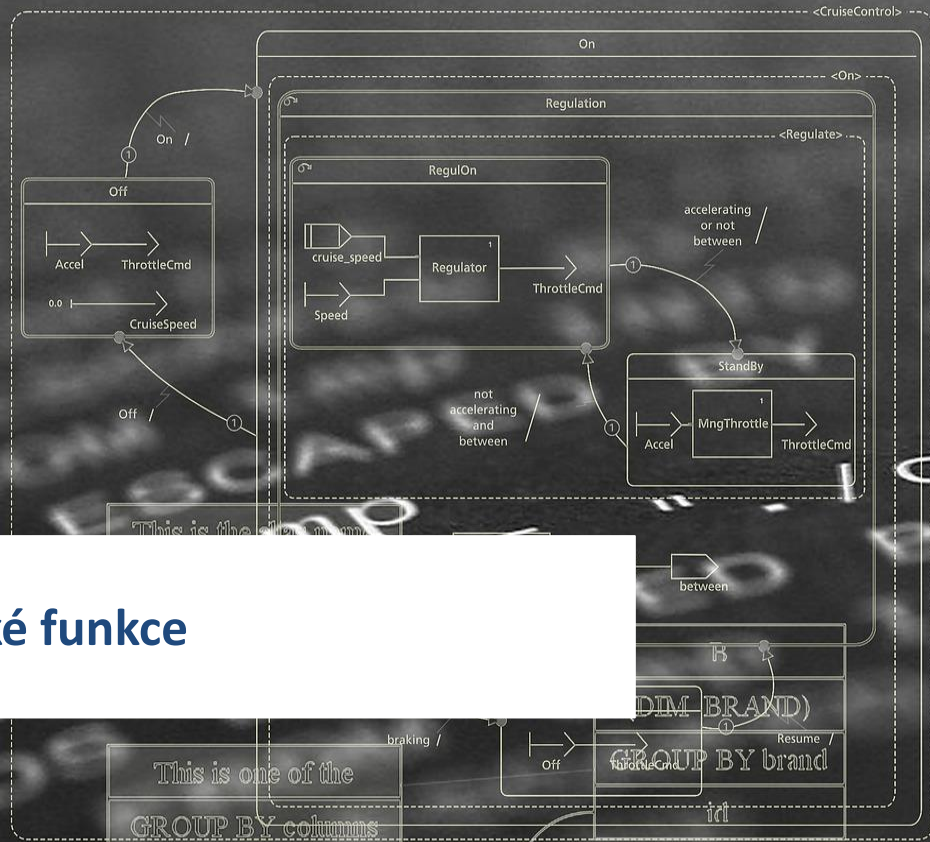
```
count(b.store_id) as poradi,  
a.store_id  
from store a join store b on (a.store_id <= b.store_id)  
group by a.store_id  
order by 1
```

```

4780 GOTO 5000
4790 :
4800 REM -----
4801 REM --- DARSTELLUNG ---
4802 REM --- DES MANUALS ---
4803 REM -----
4810 :
4820 PRINT"00";
4825 W=V+1:IF W<0 THEN W=W+14
4830 FOR X=1 TO 2:PRINT"000000000000";
4835 FOR I=0 TO 23
4840 PRINT MD$(I+W);
4850 NEXT:PRINT:NEXT
4860 PRINT"000000000000";
4870 FOR I=0 TO 23
4880 IF MD$(I+W)=CHR$(32) THEN PRINT MB$
(I+1);:GOTO 4900
4890 PRINT MD$(I+W);
4900 NEXT
4910 PRINT:PRINT"000000000000";
4920 FOR I=2 TO 24 STEP 2
4925 PRINT"|";
4930 IF MD$(I+W-1)=
";:GOTO 4940
4935 PRINT" ";
4940 NEXT:PRINT"0"
4950 PRINT"000000000000";
4960 FOR I=2 TO 24
4965 PRINT"|";
4970 IF MD$(I+W-1)="00 00" THEN PRINT"0"
MB$(I)"0";:GOTO 4980
4975 PRINT MB$(I);
4980 NEXT:PRINT"0"

```

Analytické funkce



Agregační funkce

```
select count(*) from title
```

```
select count(price) from title
```

```
select count(distinct price) from title
```

- Počet řádků
- Počet nenulových hodnot price
- Počet různých nenulových hodnot price

Analytické funkce

CORR(<expression1>, <expression2>) OVER (<analytic clause>)

COVAR_POP(<expression1>, <expression2>) OVER (<analytic clause>)

COVAR_SAMP(<expression1>, <expression2>) OVER (<analytic clause>)

CUME_DIST(<value>) OVER (<partition_clause> <order by clause>)

DENSE_RANK() OVER (<query_partition_clause> <order_by_clause>)

FIRST_VALUE(<expression> [IGNORE NULLS]) OVER (<analytic clause>)

LAG(<value expression>, <offset>, <default>) OVER ([<query partition clause>]
<order_by_clause>)

LAST_VALUE (<expression> IGNORE NULLS) OVER (<analytic clause>)

LEAD(<expression, offset, default>) [(<query_partition_clause>)] OVER
(<order_by_clause>)

NTILE (<expression>) OVER ([query_partition_clause] <order by clause>)

PERCENT_RANK(<value>) OVER (<partition_clause> <order_by_clause>)

Analytické funkce

PERCENTILE_CONT(<value>) WITHIN GROUP (ORDER BY <expression> [ASC | DESC])
OVER (<partition_clause>)

PERCENTILE_DISC(<expression>) WITHIN GROUP (ORDER BY <order_by_clause>)

RANK(<value>) OVER (<partition_clause> ORDER BY <order_by_clause>)

RATIO_TO_REPORT(<value>) OVER (<partition_clause>)

ROW_NUMBER(<value>) OVER (<partition_clause> ORDER BY <order_by_clause>)

STDDEV([DISTINCT | ALL] <expression>) OVER (<analytic_clause>)

STDDEV_POP(<expression>) OVER (<analytic_clause>)

STDDEV_SAMP(<expression>) OVER (<analytic_clause>)

VAR_POP(<value>) OVER (<analytic_clause>)

VAR_SAMP(<value>) OVER (<analytic_clause>)

VARIANCE([DISTINCT | ALL] <value>) OVER (<analytic_clause>)

Count – analytická funkce

V kolika různých typech knih se vyskytuje kniha se stejnou cenou:

TITLE_ID	TYPE	PRICE	TYPE_COUNT
-----	-----	-----	-----
BU2075	business	2.99	2
BU1032	business	19.99	3
BU7832	business	19.99	3
BU1111	business	11.95	2
MC3021	mod_cook	2.99	2
MC2222	mod_cook	19.99	3
PC1035	popular_comp	22.95	1
PC9999	popular_comp		2
PC8888	popular_comp	20	1
PS2091	psychology	10.95	1
MC3026	psychology		2

Řešení

select

title_id,

type,

price,

count(distinct type) **over (partition by price) as**
type_count

from title

order by type;

Řazení – číslování řádků

Seřadíte knihy podle prodeje.

ORDER	TOTAL_SALES	TYPE	TITLE_ID
1	111	psychology	PS2106
2	375	psychology	PS1372
3	375	trad_cook	TC3218
4	2032	mod_cook	MC2222
5	2045	psychology	PS2091
6	3336	psychology	PS7777
7	3876	business	BU1111
8	4072	psychology	PS3333
9	4095	trad_cook	TC7777
10	4095	popular_comp	PC8888
11	4095	business	BU1032
12	4095	business	BU7832
13	8780	popular_comp	PC1035
14	15096	trad_cook	TC4203

Řešení

select

row_number()

over (ORDER BY TOTAL_SALES) as "ORDER",

total_sales,

type,

title_id

from

title

Rozhodování - case

Skryjte opakující se hodnoty.

ORDER	TOTAL_SALES	TYPE	TITLE_ID
1	111	psychology	PS2106
2	375	psychology	PS1372
3	375	trad_cook	TC3218
	2032	mod_cook	MC2222
	2045	psychology	PS2091
5	3336	psychology	PS7777
6	3876	business	BU1111
7	4072	psychology	PS3333
8	4095	trad_cook	TC7777
9	4095	business	BU7832
	4095	business	BU1032
	4095	popular_comp	PC8888
	8780	popular_comp	PC1035
10	15096	trad_cook	TC4203

Řešení

select

```
case LAG("ORDER") over (order by "ORDER")  
when "ORDER" then null  
else "ORDER" end as order,  
total_sales,  
type,  
title_id
```

from

```
(select dense_rank() over (order by total_sales) as  
"ORDER", total_sales,    type, title_id from title)
```

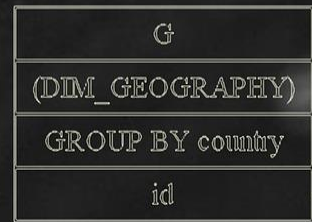
Závěr

- SQL jazyk je velice proprietární
- Obsahuje veliké množství specialit spojených s jednotlivými datovými servery
- Psaní obecně použitelného kódu nedává moc smysl
 - Dosažení platformové nezávislosti (například repository nějakého systému) vysoce omezuje použité konstrukce
 - Různé typy interface mohou částečně pomoci

Co si zapamatovat

- Na co je potřeba dát pozor při práci s hodnotami typu datum
- Základní části příkazu select
- Jaký je rozdíl mezi implementací rostoucí řady pomocí identity a sekvencí
- Co jsou základní chyby při práci s null hodnotou
- Jaké typy operace join existují, na co si je potřeba dát pozor při jejich použití
- Jak a k čemu se používá klauzule with
- Co to jsou analytické funkce, k čemu slouží

This is a WHERE clause



Diskuse

- Otázky
- Poznámky
- Komentáře
- Připomínky

